Comp15
Project 1

Simona Rolli

Introduction

- The problem:
  - A list is given of employees
    - Id
    - Name, Role
    - Manager_id
    - Salary
  - Design a data structure such that the following queries will be answered in an optimized way:

Employee ID	First Name	Last Name	Designation	Manager ID	Salary
296754708	Smith	Wendy	sales engineer	43051499	78822
375393913	Yates	Bertha	frontline contact	43051499	68356
449416814	Reed-Moore	Eileen	sales engineer	43051499	68548
605793341	Hoffman	Sarah	frontline contact	43051499	51909
721680952	Norman-Welch	Sarah	sales engineer	43051499	64193
103153280	Thompson	Norma	project manager	79916626	116090
109998233	Henderson	Diane	project manager	79916626	142580
779228460	Curry	Roberta	project manager	79916626	131777
780196582	Gilbert	Gladys	project manager	79916626	112852

1. What is the maximum salary and who has this salary?
2. What is the minimum salary and who has this salary?
3. Which employees have salaries in the range (x, y)?

4. Given an Employee ID, find who is his/her Manager?
5. Given an Employee ID, find the chain of command? You should return the name and designation of all the employees in the chain. Everyone has a boss except the president (look at the data).

Important variables

- The variables important to the problem are:
  - employee ID
  - Manager ID

  } Chain of command

  - salary → Salary inquiries

Chain of command

- For each employee there is one and only one manager
  - For each manager there are possibly several employees;
  - The CEO does not have a manager ;
- Each “employee” object should contain the pointer to its manager: linked list.
  - Traversal time linear O(N)

Salary queries

- The salary queries would be the fastest if the salaries of the employees would be ordered:
  - Suppose we order them in decreasing order:
    - Max salary: salary(employee[0]) constant time
    - Min salary : salary (employee[size -1]) constant time
    - Salary within range: binary search log(N) time

Implementation

- Object Employee

```
Struct Employee {
Employee* manager;
int ID;
Int manager_ID;
int salary;
string name;
string role;

}
```

- Vector of employees where we sort the elements by salary

```
Vector<Employee> v ;
```

Implementation (cont'd)

1. read the input from file O(N)

```
while (infile != end){
Employee * node;
Fill the data fields of Employee
Leave the pointer NULL
v.push_back(*node);
```

2. sort the elements of the vector by their salary $O(N^2)$

```
for (int j = 0 ; j < v.size(); j++) {
      for (int k = j + 1; k < v.size(); k++ ){
          if (v[k].salary > v[j].salary){
             Employee temp = v[k];
             v[k] = v[j];
             v[j] = temp;
             } // if
           } // for k
          } //  for j
```

Implementation (cont'd)

3. For each element of the vector, set its pointer to the manager

```
    // for each j I check against all the k
    // I will find one manager for each j and I'll set the
    // pointer in v[j] to v[k]

for (int j = 0 ; j < v.size() ; j++) v[j].internal_id  = j;
          for (int k = 0; k < v.size(); k++) {

             if(v[j].manager_id == v[k].id && v[j].manager_id != v[j].id){
             v[j].manager = &v[k] ;
             v[j].internal_idref = v[k].internal_id; // here idref points to the internal_id

      } // if(v[j].manager_id == v[k].id )

    } //for (int k = 0; k < v.size(); k++)
} // for (int k = 0; k < v.size(); k++)
```

Check on the ceo

All can be done with Indexes instead of pointers

$O(N^2)$

Queries (cont'd)

- Given an employee name, find the chain of command:

```
vector<Employee>::iterator Itr;
 // loop on all employee and find one corresponding the a name
 for (Itr = v.begin(); Itr!= v.end(); Itr++) {
   if (Itr->lastName == "Galli" && Itr->firstName == "Federica" ){
     if (Itr->manager == 0 ) {
         std::cout << " found CEO " << std::endl;
     }

     else { // if Itr.manager_id !=0
         Employee* temp = &(*Itr);
         string temp_lastName  = Itr->lastName;
         string temp_firstName = Itr->firstName;
         int temp_manID = Itr->internal_idref;
         int i = 0;

         while ( temp != 0) {
           i++;
           std::cout << " The manager of level " << i <<  " of "
                       << temp_lastName << " " << temp_firstName <<   " is "
                       << temp->lastName << " " << temp->firstName <<  std::endl;
                     temp = temp->manager;
                   } // while
     }// else
   } // if
 } // for Itr
```

$O(N^2)$

Queries (cont'd)

- `Max salary: v[0].salary;`
- `Min salary: v[v.size()-1].salary;`
- `Salary within range a-b : binary search?`

Const time

Log(N)